نتطرق في هذا الفصل إلى مجموعة من الأوامر التي من الممكن استعمالها في التدريس، أو نحتاجها خلال عملنا بهذا البرنامج، وزودنا هذه الأوامر بمجموعة من الأمثلة، للتوضيح ولتسهيل الفهم، وراعينا قدر المستطاع التدرج من البسيط إلى المعقد، وهذه الأمثلة قمنا بنقلها من برنامج sage بواجهة مستخدم"shell" بواسطة الأمر "copier".

يستعمل برنامج sage مثل جميع البرامج الأخرى معظم الرموز المتعارف عليها في الرياضيات، من رموز الدوال والمعادلات، وبعض الأوامر.

ونتطرق الآن إلى مجموعة من الأوامر والأمثلة لتبسيط عمل البرنامج للمستخدم.

I. أمثلة عن العمليات الحسابية:

1.I. الجمع:

نستعمل الرمز "+" ندخل العملية الحسابية في الخلية ثم نضغط على shift-enter.

```
sage: 2+7
9
```

2.I. الطرح:

نستعمل الرمز "–".

```
sage: 7-2
5
sage: 2-7
-5
```

3.I. الضرب:

نستعمل الرمز "*".

```
sage: 2*7
14
```

4.I. القسمة:

نستعمل من أجل القسمة الرمز "/" وبواسطة هذه العملية يقوم sage بإيجاد أبسط كسر، لنتفقد الأمثلة التالية.

1- قسمة عدد طبيعي على آخر يقبل القسمة عليه:

```
sage: 80/4
20
```

2- قسمة عددين أوليين فيما بينهما:

```
sage: 125/4
125/4
```

3- قسمة عددين غير أوليين فيما بينهما:

```
sage: 16/12
4/3
```

ولإيجاد النتيجة بالقيمة العشرية نستعمل الأمر "n()." ، أو بإضافة نقطة إلى العدد ليدل على أنه عدد عشري.

```
sage: n(16/12)
1.33333333333333
```

ومن أجل عدد الأرقام (الدقة) نضيف الأمر"digits=a"، حيث a يمثل عدد أرقام النتيجة.

```
sage: n(16/12,digits=5),16/12.n(digits=19)
(1.3333, 1.333333333333333333)
```

5.I. الرفع إلى قوة:

من أجل رفع عدد إلى قيمة معينة نختار أحد الرمزين التاليين "**" أو "^".

ملاحظة: من أجل إدخال عمليتين في سطر واحد نضيف الرمز ";" بين العمليتين، لتظهر نتيجة كل عملية في سطر منفصل أو باستعمال الرمز "," من أجل تنفيذ العمليتين في سطر واحد.

```
sage: 123**16,123^2
(27446172150134448422487232424420161, 15129)
sage: 123**16;123^2
27446172150134448422487232424420161
15129
```

.6.I الكتابة العلمية:

نستعمل الرمز "e" من أجل كتابة قوة 10 مثلا جمع $10^{32} + 10^{15}$.

```
sage: 123e32/5e15
2.46000000000000e18
```

ماذا يحدث إذا كتبنا الأمر التالي $2^{**}(-4)$ ؟

نترك الإجابة لكم.

.7.I العاملي:

من أجل إيجاد عاملي عدد!n نستعمل الأمر"factorial(n)"

```
sage: factorial(10),factorial(9),10*factorial(9)
(3628800, 362880, 3628800)
sage: factorial(0)
1
```

.8.I حساب المجاميع:

$$\sum_{x=a}^{n} f(x)$$

نستعمل الأمر "sum()" ونستعمله بالطريقة التالية.

$$\sum_{x=a}^{n} f(x) = sum(f(x)\ for\ x\ in\ (a..n))$$

```
sage: sum(x^2+2*x for x in (1..20))
3290
```

.9.I الجداءات:

من أجل حساب جداءات f(x) نستعمل الأمر "prod()".

$$\prod_{x=a}^{n} f(x) = prod(f(x)\ for\ x\ in\ (a..n))$$

```
sage: prod(i for i in (5..9))
15120
```

10.I. حقول الأعداد:

رموز حقول الأعداد في sage:

الأعداد الصحيحة (intgers) « ZZ »

الأعداد المنطقية (rationals) « QQ»

الأعداد الحقيقية (reals) «RR»

الأعداد التخيلية (complex) «CC»

نسمي متغير ونعطيه قيمة معينة ثم نقوم بمجموعة من العمليات عليه.

```
sage: a=5
sage: a.base_ring()
Integer Ring
sage: type(a)
<type 'sage.rings.integer.Integer'>
```

معرفة حقول الأعداد التي ينتمي إليها(a)

```
sage: a in ZZ, a in QQ , a in RR , a in CC
(True, True, True, True)
```

نعطي الآن قيمة أخرى لـ a ثم نختبر المجموعة التي ينتمي إليها العدد a من جديد

```
sage: a=4/3
sage: a.base_ring()
Rational Field
sage: type(a)
<type 'sage.rings.rational.Rational'>
sage: a in ZZ, a in QQ , a in RR , a in CC
(False, True, True, True)
```

11.I. الجذور:

الجذر التربيعي لعدد يكتب على الشكل التالي"sqrt(x)"

```
sage: sqrt(5)
sqrt(5)
sage: sqrt(112),sqrt(112).n()
(4*sqrt(7), 10.5830052442584)
```

إذا ماهي قيمة الجذر التالي$\sqrt{-1}$؟

الجذر من الدرجة n يكتب من الشكل= $\sqrt[n]{x}$ (1/n)

```
sage: 8^(1/3)
2
sage: 64**(1/4)
2*4^(1/4)
```

II. الكسور والأعداد العشرية:

1.II. مجموع و طرح كسرين يعطي كسراً بسيطاً.

```
sage: 8/3+5/2;8/3-5/2
31/6
1/6
```

2.II. مجموع أو طرح عددين عشريين يعطي عدداً عشرياً، أي أن جمع مقدارين من نفس الصنف يعطي نتيجة من نفس الصنف.

```
sage: 1.23+1.5,1.23-1.5
(2.73000000000000, -0.270000000000000)
```

فماذا يحدث إن جمعنا عددين من صنفين مختلفين؟

اجمع عدد كسري مع عدد عشري.

3.II. لإيجاد مقام كسر نستعمل الأمر"denominator()".

```
sage: a=73/55
sage: a.denominator(),denominator(a)
(55, 55)
```

4.II لإيجاد بسط كسر نستعمل الأمر"numerator()".

```
sage: a.numerator();numerator(a)
73
73
```

5.II لتدوير عدد عشري إلى العدد الصحيح الأقرب إليه نستعمل الأمر"round()"

```
sage: 4.53.round()
5
sage: 4.23.round()
4
```

6.II من أجل إيجاد الحد الأدنى لعدد عشري"floor()".

```
sage: 4.53.floor();floor(4.53)
4
4
```

7.II من أجل إيجاد الحد الأعلى لعدد عشري"ceil()".

```
sage: 4.53.ceil()
5
```

8.II العلاقات الرياضية الموجودة بين الأعداد.

أقل من " < "

```
sage: 5<6
True
```

أكبر من " > "

```
sage: 4>5
False
```

أكبر من أو يساوي " >= "

```
sage: 12>12,12>=12
(False, True)
```

رمز المساواة "=="

```
sage: 16==8*2
True
```

لا يساوي (≠) " =!"

```
sage: 15!=15,15!=12
(False, True)
```

يختلف عن " <> "

```
sage: 14<>14,14<>10
(False, True)
```

9.II في برنامج sage الخطأ يكون دائما في السطر الأخير

الخطأ

```
sage: a=4;b=6
sage: a<b,a>b,a=b
------------------------------------------------------------
  File "<ipython console>", line 1
SyntaxError: can't assign to comparison (<ipython console>, line 1)
sage: a<b,a>b,a==b
(True, False, False)
```

← رسالة الخطأ

← تصحيح الخطأ

نستعمل الرمز " = " للتصريح بقيم المتغيرات و الرمز " == " للمعادلات والمقارنة.

.III الأعداد الأولية:

1.III التحليل: لتحليل عدد إلى جداء عوامل أولية نستعمل الأمر "factor()"

```
sage: 2940.factor();factor(2940)
2^2 * 3 * 5 * 7^2
2^2 * 3 * 5 * 7^2
sage: 2**2*3*7**2*5
2940
```

2.III لمعرفة ما إذا كان العدد أوليا نستعمل الأمر "is_prime()"

```
sage: 2010.is_prime();is_prime(2011)
False
True
```

3.III لمعرفة ما إذا كان العدد هو قوة لعدد أولي نستعمل الأمر is_prime_power() "

```
sage: a=25;a.is_prime_power()
True
sage:is_prime_power(50)
False
```

لدينا $50=2\times5^2$ أي أن 50 هو قوة لعددين أوليين وليس قوة لعدد أولي واحد.

4.III لمعرفة العدد الأولي الموالي لعدد نستعمل الأمر "next_prime() "، وللعدد الأولي الذي قبل عدد نستعمل الأمر "previous_prime() "

```
sage: next_prime(4),previous_prime(4)
(5, 3)
sage: next_prime_power(16),previous_prime_power(66)
(17, 64)
```

العدد 17 هو عدد أولي وفي نفس الوقت هو قوة لنفسه لأن" $17 = 17^1$ "

```
sage: factor(64)
2^6
```

العدد 64 هو القوة السادسة للعدد الأولي 2

ما هو العدد الأولي الموالي للعدد (-7) ؟

5.III لمعرفة عدد الأعداد الأولية الموجودة قبل العدد x نستعمل الأمر "pari(x).primepi()"

```
sage: pari(20).primepi()
8
```

ولإيجاد قائمة هذه الأعداد نستعمل الأمر " list(primes(a,x))" ، الذي يقوم بإيجاد الأعداد الأولية المحصورة بين a و x

```
sage: list(primes(1,20))
[2, 3, 5, 7, 11, 13, 17, 19]
```

ولإيجاد قائمة الأعداد الأولى من الأعداد الأولية نستعمل الأمر "primes_first_n()"

```
sage: primes_first_n(10)
[2, 3, 5, 7, 11, 13, 17, 19, 23, 29]
```

IV. القواسم والمضاعفات:

1.IV لإيجاد قواسم عدد طبيعي نستعمل الأمر "divisors()".

```
sage: 2012.divisors()
[1, 2, 4, 503, 1006, 2012]
```

2.IV ولإيجاد مجموع قوى القواسم نستعمل الأمر " sigma(x,n) "، أي إذا كان k هو أحد قواسم العدد x فإننا نجمع هذه القواسم مرفوعة إلى القوة n ($\sum k^n$).

```
sage: sigma(72,0),sigma(72,1),sigma(72,2)
(12, 195, 7735)
```

3.IV باقي وحاصل القسمة:

كل عدد صحيح يكتب من الشكل a=b.c+r، حيث أن a يمثل المقسوم، وb القاسم، والعدد c حاصل القسمة.

وr هو باقي القسمة.

وللحصول على حاصل القسمة نستعمل الرمز " // " ، وللباقي نستعمل أحد الرمزين " %" أو "mod()".

```
sage: a=94;b=8
sage: c=a//b;c
11
sage: r=a%b;r;a.mod(b)
6
6
```

```
sage: a==b*c+r;b*c+r
True
94
sage: -27//5
-6
sage: -27//5;-27%5
-6
3
sage: -6*5+3
-27
sage: (-27).mod(5)
3
```

4.IV. القواسم و المضاعف المشترك لعددين:

لإيجاد القاسم المشترك الأكبر لعددين نستعمل الأمر " gcd()"، ولإيجاد المضاعف المشترك الأصغر نستعمل الأمر " lcm() "

```
sage: 15.gcd(25),15.lcm(25)
(5, 75)
sage: 5*75,25*15
(375, 375)
```

V. الأعداد المركبة (العقدية):

المجموعة ℂ تتضمن بداخلها المجموعة ℝ، وتحتوي على العدد التخيلي $i^2 = -1$

```
sage: i=CC(i)
sage: type(i)
<type 'sage.rings.complex_number.ComplexNumber'>
sage: i^2
-1.00000000000000
sage: A=5+2*i;B=3+7.2*i;A;B
5.00000000000000 + 2.00000000000000*I
3.00000000000000 + 7.20000000000000*I
```

```
sage: A+B;A*B
8.00000000000000 + 9.20000000000000*I
0.600000000000000 + 42.0000000000000*I
```

1.V الجزء الحقيقي والتخيلي:

كل عدد مركب يكتب من الشكل $z = a + b.i$

ولمعرفة الجزء الحقيقي (a) نستعمل الأمر " real() "، وللجزء التخيلي الأمر " img() "

```
sage: A=5+2*i
sage: a=real(A);b=imag(A);a,b
(5.00000000000000, 2.00000000000000)
```

2.V مرافق عدد عقدي:

لكل عدد عقدي z مرافق $\bar{z} = a - b.i$ ولإيجاد المرافق نستعمل الأمر "conjugate()"

```
sage: A.conjugate()
5.00000000000000 - 2.00000000000000*I
```

معيار العدد العقدي z ($|z| = \sqrt{z\bar{z}} = \sqrt{a^2 + b^2}$) يمكن إيجاده باستعمال الأمر"abs()"،

3.V ناظم عدد عقدي:

من أجل الناظم نستعمل الأمر "norm()"

```
sage: A.abs();A.norm(),abs(A)^2
5.38516480713450
(29.0000000000000, 29.0000000000000)
```

4.V عمدة عدد عقدي:

ولإيجاد α عمدة العدد العقدي z نستعمل الأمر "arg(z)" أو "argument()"

```
sage: A.arg();A.argument()
0.380506377112365
0.380506377112365
```

والآن نتأكد من العلاقة $z = r(\cos\alpha + \sin\alpha)$ حيث r يمثل عمدة z

```
sage: A=5+2*i
sage: r=A.abs();a=A.arg();r;a
5.38516480713450
0.380506377112365
sage: r*(cos(a)+i*sin(a))
5.00000000000000 + 2.00000000000000*I
```

5.V. الجذور النونية للعدد العقدي:

الجذر من الدرجة (5)

```
sage: A.nth_root(5)
1.39630726164159 + 0.106466372820269*I
```

كل الجذور الخمسة الأولى

```
sage: A.nth_root(5,all=true)
[1.39630726164159 + 0.106466372820269*I, 0.330227135579372 +
1.36086703846540*I, -1.19221566784602 + 0.734595711120759*I, -
1.06705694022837 - 0.906861921002872*I, 0.532738210853424 -
1.29506720140356*I]
```

VI. المعادلات والدوال:

1.VI. كتابة المعادلتين f و g

```
sage: f=(x^2+5*x==3*x+2)
sage: g=(5*x^3+x==5*x^2)
```

- طرح القيمة "x" من طرفي المعادلة f

```
sage: f-x
x^2 + 4*x == 2*x + 2
```

- رفع المعادلة g إلى الدرجة "2"، وطرح (3x+2) من طرفي المعادلة f

```
sage: g^2;f-(3*x+2)
(5*x^3 + x)^2 == 25*x^4
x^2 + 2*x - 2 == 0
```

- مجموعة من العمليات على كل من المعادلتين f و g

```
sage: f/g;f*g;f+g
(x^2 + 5*x)/(5*x^3 + x) == 1/5*(3*x + 2)/x^2
(x^2 + 5*x)*(5*x^3 + x) == 5*(3*x + 2)*x^2
5*x^3 + x^2 + 6*x == 5*x^2 + 3*x + 2
```

ماذا تلا حظ على هذه العمليات؟

تتم هذه العمليات طرف لطرف لكل دالة.

ولمعرفة طرف كل دالة نستعمل الأمر "right()" للطرف الأيمن، والأمر "left()" للطرف الأيسر.

```
sage: f=(x^2+5*x==3*x+2)
sage: g=(5*x^3+x==5*x^2)
sage: f.right()
3*x + 2
sage: g.left()
5*x^3 + x
sage: f.right()*g.left()
(3*x + 2)*(5*x^3 + x)
```

2.VI. النشر والتحليل:

لنشر دالة أو معادلة نستعمل الأمر " expand()"، ومن أجل التحليل إلى جداءات نستعمل الأمر "factor() "

```
sage: Z=(3*x + 2)*(5*x^3 + x)
sage: Z.expand()
15*x^4 + 10*x^3 + 3*x^2 + 2*x
sage: Z.factor()
(3*x + 2)*(5*x^2 + 1)*x
```

ملاحظة: المتغير" Z (كبير) " ليس هو المتغير " z (صغير)" لذا يجب الانتباه إلى رموز المتغيرات.

3.VI. إيجاد قيمة دالة عند نقطة:

من أجل إيجاد قيمة الدالة عند نقطة معينة نستعمل الأمر "subs() "

```
sage: Z.subs(x=5)
10710
sage: Z (x=5)
10710
```

4.VI. إيجاد جذر دالة في مجال:

في كثير من دوال الدرجة الثالثة تواجهنا مشكلة البحث عن جذر للدوال، من أجل إيجاد جذر الدالة عند مجال معين نستعمل الأمر " find_root() "

```
sage: Z.find_root(-2,-.5)
-0.66666666666666607
sage: Z.find_root(0,10)
0.0
```

5.VI. حل المعادلات:

من أجل حل المعادلات نستعمل الأمر "solve() "

```
sage: solve(x^2+5*x-7==0,x)
[x == -1/2*sqrt(53) - 5/2, x == 1/2*sqrt(53) - 5/2]
sage: solve(x^2+5*x-7,x)
[x == -1/2*sqrt(53) - 5/2, x == 1/2*sqrt(53) - 5/2]
```

المتغير x يعتبر تلقائيا كمتغير ولإضافة متغيرات أخرى يحب التصريح بها أولا بواسطة الأمر "var('y')"

```
sage: var('a')
a
sage: solve(a^2+10*a+25,a)
[a == -5]
```

حل وحيد مضاعف

كما يمكن حل المعادلة بدلالة متغيرات أخرى

```
sage: f=a*x^2+b*x+c
---------------------------------------------------------------------------
NameErrorTraceback (most recent call last)
/home/sage/<ipython console> in <module>()
NameError: name 'a' is not defined
```

كما ترى حصل هناك خطأ وهو أن 'a' غير معرف، أي اننا يجب أن نصرح بكل متغيرات أولا قبل العمل بها، ماعدا 'x' فهو متغير مصرح به في البرنامج

```
sage: var('a,b,c')
(a, b, c)
sage: f=a*x^2+b*x+c
sage: solve(f,x)
[x == -1/2*(b + sqrt(-4*a*c + b^2))/a, x == -1/2*(b - sqrt(-4*a*c + b^2))/a]
```

كما يمكن للبرنامج إيجاد حلول العمليات في المجموعة $\mathbb{C}$

```
sage: S=solve(x^2+2,x);S
[x == -I*sqrt(2), x == I*sqrt(2)]
```

الحل الأول للمعادلة

```
sage: S[0]
x == -I*sqrt(2)
```

الحل الثاني للمعادلة

```
sage: S[1]
x == I*sqrt(2)
```

كما نستطيع حل جمل المعادلات (ثلاث متغيرات مثلا):

```
sage: var('x,y,z')
(x, y, z)
sage: eq1=5*x+2*y+2*z^2
sage: eq2=2*x+3*y+z
sage: eq3=x+y+7*z
sage: solve([eq1,eq2,eq3],x,y,z)
[[x == 0, y == 0, z == 0], [x == -740, y == 481, z == 37]]
```

يجب وضع المعادلات بين عارضتين حتى يعلم البرنامج ماهي المعادلات التي يجب عليه حلها.

عندما يكون النتيجة مجموعة من الحلول تظهر كالتالي

```
sage: var('x,y')
(x, y)
sage: solve([x+y==2,2*x+2*y==4],x,y)
[[x == -r1 + 2, y == r1]]
```

تتحقق المعادلات من أجل كل قيمة لـr1

6.VI. النهايات:

لإيجاد نهاية الدالة f(x) عندما يؤول x إلى القيمة a نستعمل الأمر "limit()" ويكتب على الشكل التالي:

$$\lim_{x\to a} f(x) = \text{limit}(\text{f(x)}, \text{x} = \text{a})$$

```
sage: f=(2*x-2)^2/(x-1)
sage: limit(f(x),x=1)
0
sage: g=x+1/(x-1)
sage: limit(g(x),x=1)
Infinity
```

ولحساب النهاية عند قيم أعلى أو أدى نضيف الأمر " dir='plus' ، dir='minus' "

```
sage: limit(g ,x=1,dir='plus')
+Infinity
sage: limit(g ,x=1,dir='minus')
-Infinity
```

لحساب النهاية عند $\pm\infty$ نستعمل الحرفين "oo" باللغة الإنجليزية وللتفريق بين "0" و"o" وضعت نقطة داخل الرقم صفر

```
sage: limit(g(x),x=oo)
+Infinity
sage: limit(g(x),x=-oo)
-Infinity
```

.7.VI الاشتقاق:

من أجل اشتقاق الدوال نستعمل الأمر "diff()" أو الأمر "derivative()"

```
sage: g=5*x^6+3*x^5-7*x^3+2
```

- المشتق الأول

```
sage: diff(g)
30*x^5 + 15*x^4 - 21*x^2
```

- عدد الأوامر "diff()" يمثل درجة المشتق

```
sage: g.diff().diff()
150*x^4 + 60*x^3 - 42*x
```

أو يمكن كتابة درجة المشتق بين قوسي الأمر "diff()"

```
sage: g.diff(2)
150*x^4 + 60*x^3 - 42*x
sage: diff(g,5) ; g.diff(6)
3600*x + 360
3600
```

في الدوال ذات الأكثر من متغير يمكن حساب المشتقات الجزئية، نقوم بإدخال المتغير داخل القوسين، وهذا مثال عن المشتقات الجزئية

```
sage: q=x^5+4*y^7+5*x^4*y^3-x*y
sage: diff(q,x);diff(q,y)
20*x^3*y^3 + 5*x^4 - y
15*x^4*y^2 + 28*y^6 - x
```

أو بالنسبة للمتغيرين معا

```
sage: diff(q,x,y)
60*x^3*y^2 - 1
```

11.VI. التكامل:

أما بالنسبة لعملية التكامل نستعمل الأمر "integrate()" أو "integral() "

1) التكامل غير المحدود:

- متغير واحد

```
sage: p=2*x^2+5;p
2*x^2 + 5
sage: p.integral(x)
2/3*x^3 + 5*x
```

- معادلة بأكثر من متغير:

```
sage: var("x y z")
(x, y, z)
sage: f= x^2+2*y^3-7*z+x*y;f
x^2 + 2*y^3 + x*y - 7*z
sage: f.integral(x);f.integrate(x,y);f.integrate(z)
2*x*y^3 + 1/3*x^3 + 1/2*x^2*y - 7*x*z
-2*x*y^3 - 1/3*x^3 + 2*y^4 - 1/2*x^2*y + 5/6*y^3 + 7*x*z - 7*y*z
2*y^3*z + x^2*z + x*y*z - 7/2*z^2
sage: f.integral(x,y,z)
-2*y^4 + 2*y^3*z + 1/2*(y - 14)*z^2 - 5/6*y^3 + 1/3*z^3 + 7*y*z
```

- التكامل المحدود:

التكامل المحدود للدالة $f(x)$ من القيمة a إلى القيمة b يكتب على الشكل التالي

- متغير واحد:

$$\int_{a}^{b} f(x)dx = integrate(f,(x,a,b))$$

```
sage: p.integrate(x,2,10)
2104/3

sage: integrate(p,(x,2,10))
2104/3
```

- معادلة بأكثر من متغير:

```
sage: integral(integral(integral(f,(x,1,5)),(y,2,7)),(z,-4,3))
111650/3
```

VII. المعادلات التفاضلية:

لتوضيح كيف يمكن حل المعادلات التفاضلية نرى المثال التالي

المعادلات التفاضلية الخطية من النوع $\acute{x} + x + 1 = 0$

```
sage: var("t")
t
sage: x=function("x",t)
sage: de=diff(x,t)+x-1
sage: desolve(de,[x,t])
(c + e^t)*e^(-t)
```

إجراء تحويل لابلاس

```
sage: var("t s ")
(t, s, )
sage: f=t^2+exp(t)-sin(t)
sage: f.laplace(t,s)
1/(s - 1) - 1/(s^2 + 1) + 2/s^3
```

VIII. كثيرات الحدود:

- ✓ كتابة مجموعة كثيرات الحدود في حلقة الأعداد الناطقة.

```
sage: R.<x> = PolynomialRing(QQ)
sage: f = (x-1)^2 * (x-6)^2 * (x^2 + 1)
sage: g = (x-1)^3 * (x-6) * (3*x – 10)
```

- ✓ التأكد من نوعي الدالتين:

```
sage: type(f);type(g)
<type‘sage.rings.polynomial.polynomial_rational_flint.Polynomial_rational_flint’>
<type‘sage.rings.polynomial.polynomial_rational_flint.Polynomial_rational_flint’>
```

- ✓ إيجاد قيمة كثيري الحدود عند بعض القيم.

```
Sage: f(0),f(1);g(5),g(89)
(36, 0)
(-320, 14536479232)
```

- ✓ القيام ببعض العمليات على كثيري الحدود.

```
Sage: f+g ; f-g ; f*g ; f/g ; g/f
x^6 – 11*x^5 + 25*x^4 + 55*x^3 – 170*x^2 + 124*x – 24
x^6 – 17*x^5 + 99*x^4 – 251*x^3 + 364*x^2 – 292*x + 96
3*x^11 – 79*x^10 + 857*x^9 – 4997*x^8 + 17349*x^7 – 38361*x^6 +
57959*x^5 – 64187*x^4 + 53992*x^3 – 32904*x^2 + 12528*x – 2160
(x^3 – 6*x^2 + x – 6)/(3*x^2 – 13*x + 10)
(3*x^2 – 13*x + 10)/(x^3 – 6*x^2 + x – 6)
```

- ✓ إيجاد حاصل وباقي قسمة f على g.

```
Sage: f//g ; f%g
1/3*x – 5/9
-86/9*x^4 + 76*x^3 – 362/3*x^2 + 464/9*x + 8/3
```

- ✓ إيجاد حاصل وباقي قسمة g على f.

```
Sage: g//f ; g%f
0
3*x^5 – 37*x^4 + 153*x^3 – 267*x^2 + 208*x – 60
```

- ✓ التحليل إلى جداءات

```
sage: f.factor() ; g.factor()
(x – 6)^2 * (x – 1)^2 * (x^2 + 1)
(3) * (x – 6) * (x – 10/3) * (x – 1)^3
```

- ✓ القاسم المشترك الأكبر لكثيري الحدود.

```
Sage: f.gcd(g) ; gcd(f,g) ; g.gcd(f)
x^3 – 8*x^2 + 13*x – 6
x^3 – 8*x^2 + 13*x – 6
x^3 – 8*x^2 + 13*x – 6
```

✓ المضاعف المشترك الأصغر لكثيري الحدود.

```
sage: f.lcm(g)
x^8 - 55/3*x^7 + 126*x^6 - 1240/3*x^5 + 2185/3*x^4 - 831*x^3 + 2170/3*x^2
- 436*x + 120
```

IX. الفضاءات الشعاعية:

1.XI. إدخال شعاع:

```
sage:  a = vector([1, 2, 3]);a
(1, 2, 3)
```

شعاع في مجموعة الأعداد الناطقة عدد عناصره (8)

```
sage:  V = VectorSpace(QQ, 8); V
Vector space of dimension 8 over Rational Field
sage: A1=V([1/5,5,17,94,27,2/5,5,3/8])
sage: A1
(1/5, 5, 17, 94, 27, 2/5, 5, 3/8)
```

2.XI. بعد شعاع: من أجل معرفة بعد شعاع نستعمل الأمر "dimension()"

```
sage: V.dimension()
8
```

3.XI. ومن اجل معرفة قاعدة الشعاع نستعمل الأمر "basis()"

```
sage: V.basis()
[
(1, 0, 0, 0, 0, 0, 0, 0),
(0, 1, 0, 0, 0, 0, 0, 0),
(0, 0, 1, 0, 0, 0, 0, 0),
(0, 0, 0, 1, 0, 0, 0, 0),
(0, 0, 0, 0, 1, 0, 0, 0),
(0, 0, 0, 0, 0, 1, 0, 0),
(0, 0, 0, 0, 0, 0, 1, 0),
(0, 0, 0, 0, 0, 0, 0, 1)
]
```

قاعدة شعاع على شكل مصفوفة

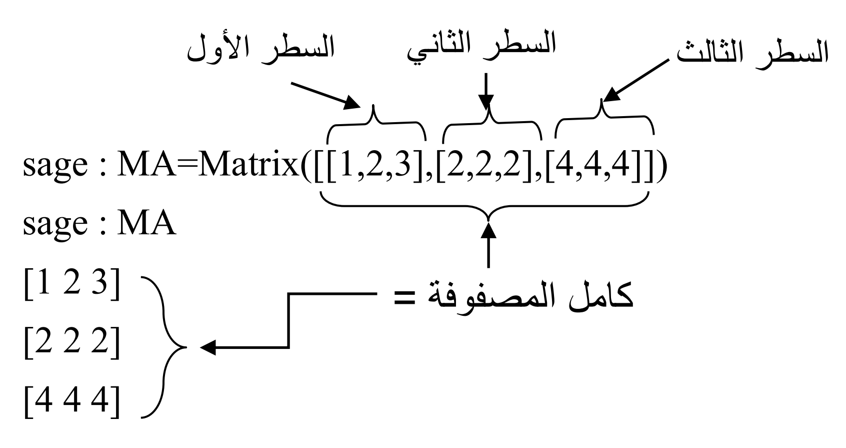
```
sage: V.basis_matrix()
[1 0 0 0 0 0 0 0]
[0 1 0 0 0 0 0 0]
[0 0 1 0 0 0 0 0]
[0 0 0 1 0 0 0 0]
[0 0 0 0 1 0 0 0]
[0 0 0 0 0 1 0 0]
[0 0 0 0 0 0 1 0]
[0 0 0 0 0 0 0 1]
```

4.XI. درجة شعاع:

```
sage: A1.degree()
8
```

5.XI. المصفوفات:

إدخال مصفوفة في برنامج sage سهلة، فبواسطة الأمر"Matrix()" ندخل المصفوفة و نضع عناصر كل سطر بين عارضتين ونضع الكل بين عارضتين أخرتين، لاحظ المثال التالي:



6.XI. إنشاء مصفوفة من مجموعة أشعة :

```
sage: v1=vector([1,5,9,8])
sage: v2=vector([2,-5,19,3])
sage: v3=vector([21,4,6,0])
sage: H=matrix([v1,v2,v3])
```

```
sage: H
[ 1  5  9  8]
[ 2 -5 19  3]
[21  4  6  0]
```

7.XI. مصفوفة الوحدة:

```
sage: I = identity_matrix(3)
sage: I
[1 0 0]
[0 1 0]
[0 0 1]
```

8.XI. المصفوفة العشوائية: ننشئ مصفوفة عشوائية بواسطة الأمر "random_matrix()".

كتابة مصفوفة عشوائية 3x3 في مجموعة الأعداد الحقيقية.

```
sage: m= random_matrix(RR,3)
sage: m
[-0.632577740413993   0.562020491802490   0.495950374230578]
[-0.440932328303367   0.184753155439922   0.862081369288889]
[-0.655681068642257  0.0261653265207265  -0.524430877925466]
```

مصفوفة عشوائية 3x3 في مجموعة الأعداد الناطقة.

```
sage: m = random_matrix(QQ,3)
sage: printm.str()
[  0   0   0]
[  0  -2 1/2]
[  0   2   0]
```

كلما أعدنا كتابة صيغة المصفوفة العشوائية m تتغير قيمتها.

```
sage: m = random_matrix(QQ,3)
sage: printm.str()
[   0   -1   -2]
[  -1    0    0]
[-1/2    0   -2]
```

9.XI. إضافة عنصر إلى مصفوفة:

```
sage: A = identity_matrix(4)
sage: A.add_multiple_of_row(0, 3, 100); A
[ 1 0 0 100]
[ 0 1 0 0]
[ 0 0 1 0]
[ 0 0 0 1]
```

10.XI. نواة مصفوفة:

لإيجاد نواة المصفوفة (MA) التي أدخلناها في بداية الموضوع نستعمل الأمر "kernel()" وبواسطة هذا الأمر نجد درجة المصفوفة، ورتبتها، ونواتها.

```
sage:  kernel(MA)
Free module of degree 3 and rank 1 over Integer Ring
Echelon basis matrix:
[ 0  2 -1]
```

11.XI. صورة مصفوفة:

```
sage: MA.image()
Free module of degree 3 and rank 2 over Integer Ring
Echelon basis matrix:
[ 1  0 -1]
[ 0  2  4]
```

محدد مصفوفة:

```
sage: det(MA)
0
```

12.XI. رتبة مصفوفة:

```
sage: MA.rank()
2
```

13.XI. جمع مصفوفتين:

المصفوفة A ذات الثلاثة أسطر والأربعة أعمدة مكتوبة في مجموعة الأعداد الناطقة ضمن المجال (3،15).

```
sage: A=matrix(QQ,3,4,range(3,15));A
[ 3  4  5  6]
[ 7  8  9 10]
[11 12 13 14]
```

عند التصريح ببعد المصفوفة برقم واحد فهذا يعني أنها مصفوفة مربعة.

```
sage: B=matrix(QQ,3,range(12));B
[ 0  1  2  3]
[ 4  5  6  7]
[ 8  9 10 11]
sage: C=A+B;C
[ 3  5  7  9]
[11 13 15 17]
[19 21 23 25]
```

14.XI. ضرب مصفوفة في عدد:

```
sage: var("a")
sage: a*C
[ 3*a  5*a  7*a  9*a]
[11*a 13*a 15*a 17*a]
[19*a 21*a 23*a 25*a]
```

15.XI. ضرب مصفوفتين:

```
sage: D=matrix(ZZ,4,4,range(16))
sage: D
[ 0  1  2  3]
[ 4  5  6  7]
[ 8  9 10 11]
[12 13 14 15]
```

```
sage: E=C*D
sage: E
[184 208 232 256]
[376 432 488 544]
[568 656 744 832]
```

16.XI. رفع مصفوفة إلى قوة:

```
sage: D^5
[ 1780800  2050000  2319200  2588400]
[ 5147200  5925200  6703200  7481200]
[ 8513600  9800400 11087200 12374000]
[11880000 13675600 15471200 17266800]
```

17.XI. مقلوب مصفوفة:

من أجل إيجاد مقلوب مصفوفة نرفعها إلى (-1)، أو نستعمل الأمر "inverse()"

ملاحظة: إذا لم يكن السطر كافيا لكتابة كل المعلومات اللازمة فإننا نضع في آخر السطر الرمز "\" ثم نضغط "enter" لنكمل الكتابة في السطر الجديد.

```
sage: dd=matrix(4,4,[-1,-1,-1,3,-1,2,-2,1,1\
....: ,0,1,3,1,-7,-11,21]);dd
[ -1  -1  -1   3]
[ -1   2  -2   1]
[  1   0   1   3]
[  1  -7 -11  21]
sage: dd^(-1)
[-157/216    -1/18     7/54   19/216]
[   -5/36      1/3      2/9    -1/36]
[    8/27     -1/9     7/27    -2/27]
[  31/216     1/18    11/54   -1/216]
sage: dd.inverse()
[-157/216    -1/18     7/54   19/216]
[   -5/36      1/3      2/9    -1/36]
[    8/27     -1/9     7/27    -2/27]
[  31/216     1/18    11/54   -1/216]
```

18.XI. منقول مصفوفة:

```
sage: dd.transpose()
[ -1  -1   1   1]
[ -1   2   0  -7]
[ -1  -2   1 -11]
[  3   1   3  21]
```

19.XI. أعمدة وأسطر مصفوفة:

من أجل إيجاد الأسطر نستعمل الأمر"rows()"، ومن أجل الأعمدة نستعمل الأمر "columns()".

```
sage: M=random_matrix(ZZ,4,3);M
[ 0  0 -1]
[23  0  1]
[-1 -1 -1]
[ 3 -1 -1]
sage: a=M.rows();a
[(0, 0, -1), (23, 0, 1), (-1, -1, -1), (3, -1, -1)]
sage: a[1]
(23, 0, 1)
sage: a[0]
(0, 0, -1)
```

أو يمكن إيجاد الأسطر بالطريقة التالية:

```
sage: M[0]
(0, 0, -1)
sage: M[1]
(23, 0, 1)
```

إيجاد العنصر الموجود في السطر الثاني العمود الأول

```
sage: M[1,0]
23
```

ومن أجل إعطاء قائمة بأعمدة المصفوفة نكتب

```
sage: b=M.columns();b
[(0, 23, -1, 3), (0, 0, -1, -1), (-1, 1, -1, -1)]
sage: b[0];b[1]
(0, 23, -1, 3)
(0, 0, -1, -1)
```

20.XI. إيجاد الشعاع x في المعادلة $A * x = v$:

يمكـــن حـــل جملـــة المعـــادلات التاليـــة بواســـطة المصـــفوفات:

$$\begin{cases} 4x + 3y - z = 3 \\ 2x + 2y + 3z = 2 \\ -x + y + z = 0 \end{cases}$$

يمكن كتابة جملة المعادلات بالطريقة التالية:

$$\begin{pmatrix} 4 & 3 & -1 \\ 2 & 2 & 3 \\ -1 & 1 & 1 \end{pmatrix}\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 3 \\ 2 \\ 0 \end{pmatrix}$$

نحل المعادلة بواسطة الأمر "m.solve_right(v)"

```
sage: m=matrix(3,[4,3,-1,2,2,3,-1,1,1]);v=vector([3,2,0])
sage: X = m.solve_right(v)
sage: X
(11/23, 9/23, 2/23)
```

وإذا تم كتابة جملة المصفوفات على الشكل $XA = V$:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix}\begin{pmatrix} 4 & 3 & -1 \\ 2 & 2 & 3 \\ -1 & 1 & 1 \end{pmatrix} = \begin{pmatrix} 3 \\ 2 \\ 0 \end{pmatrix}$$

نحل المعادلة بواسطة الأمر "m.solve_left(v)"

```
sage: X=m.solve_left(v)
sage: X
(13/23, 6/23, -5/23)
```

كما أنه يمكن القيام بحوالي 195 عملية على المصفوفات (حسب المصفوفة)، ولإيجاد هذه العمليات نكتب اسم المصفوفة متبوعاً بنقطة ثم نضغط على الزرtab من لوحة المفاتيح.

ثم نضغط على الزر y من أجل المتابعة

```
sage: MA=Matrix([[1,2,3],[2,2,2],[4,4,4]])
sage: MA.
Display all 195 possibilities? (y or n)
MA.BKZ                                 MA.lift
MA.LLL                                 MA.linear_combination_of_columns
MA.LLL_gram                            MA.linear_combination_of_rows
MA.abs                                 MA.list
MA.act_on_polynomial                   MA.matrix_from_columns
MA.add_multiple_of_column              MA.matrix_from_rows
MA.add_multiple_of_row                 MA.matrix_from_rows_and_columns
MA.additive_order                      MA.matrix_over_field
MA.adjoint                             MA.matrix_space
MA.antitranspose                       MA.matrix_window
MA.apply_map                           MA.maxspin
MA.apply_morphism                      MA.minimal_polynomial
MA.as_sum_of_permutations              MA.minors
MA.augment                             MA.minpoly
MA.base_extend                         MA.mod
MA.base_ring                           MA.multiplicative_order
MA.block_sum                           MA.n
MA.cartesian_product                   MA.ncols
MA.category                            MA.new_matrix
MA.change_ring                         MA.nonpivots
MA.characteristic_polynomial           MA.nonzero_positions
MA.charpoly                            MA.nonzero_positions_in_column
MA.cholesky_decomposition              MA.nonzero_positions_in_row
--More--
```

اضغط enter من أجل المزيد أو اضغط backspace (زر الحذف بالتراجع) من أجل الإنهاء.

X. الرسومات البيانية:

يمكن لـ sage بواجهتيه إنشاء الرسومات البيانية ثنائية البعد، وثلاثية البعد، بكل سهولة بالإضافة إلى مجموعة من الخيارات التي من الممكن إضافتها إلى الرسومات، وعند استعمالنا لواجهة shell يظهر الرسم في متصفح الإنترنت في صورة مستقلة، وأما عند استعمال واجهة notebook فإن الرسم يكون في نفس الصفحة.

X.1 الرسم في البعد الثنائي:

1) رسم الخط: ويتم بواسطة الأمر "line()" بالطريقة التالية

(line([(x1,y1),. . .,(xn,yn)],options)

وسوف نتعرف على بعض الخيارات في الرسم في الأمثلة الموالية.

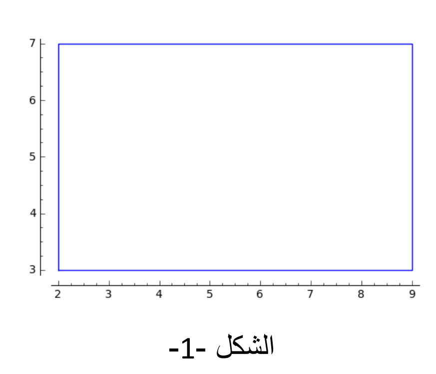
وبواسطة هذا الأمر يمكن رسم مجموعة من الأشكال الهندسية البسيطة مثلا:

- رسم مربع:

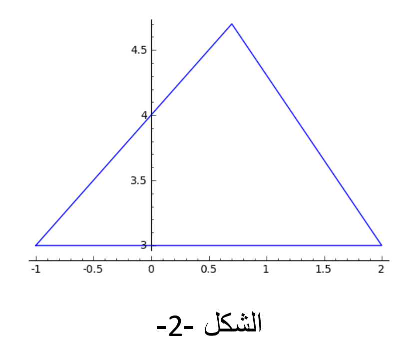
```
sage: line([(2,3),(9,3),(9,7),(2,7),(2,3)])
```

- رسم مثلث:

```
sage: line([(1-,3),(2,3),(9,7)])
```



الشكل -1-



الشكل -2-

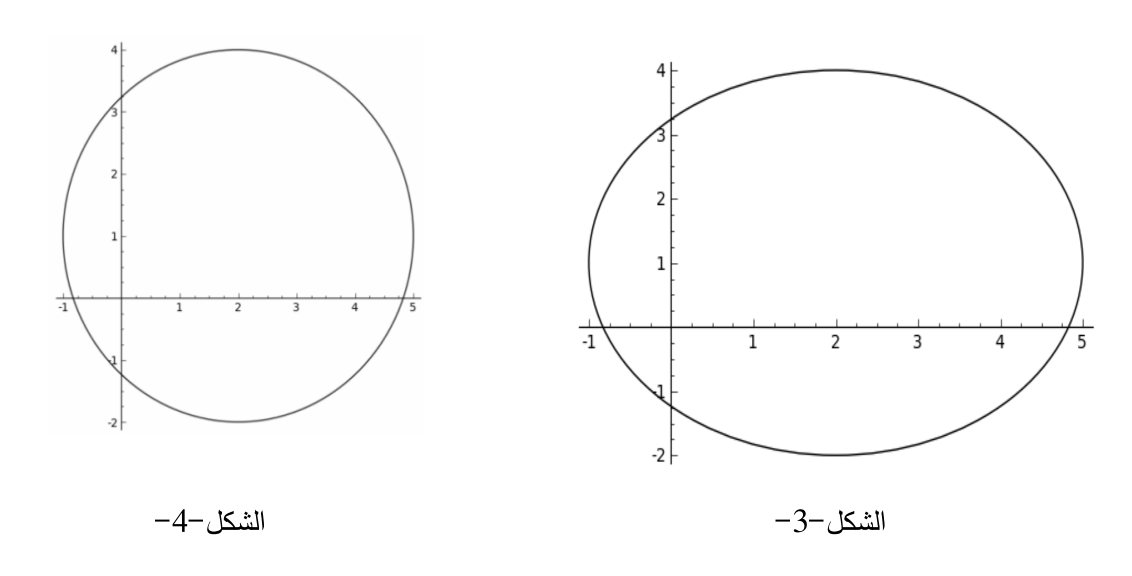
2) رسم دائرة: ويكون بواسطة الأمر "circle()" على الشكل التالي:

(Circle((x,y),r,options))

sage: circle((2,1),3)

لاحظ أن شكل (الشكل -3-) الدائرة في هذه الحالة يشبه البيضة وهذا لأن المحاور غير متجانسة ومن أجل مجانستها نضيف الخيار "aspect_ratio=1" الذي يحدد النسبة بين المحورين (الشكل-4-).

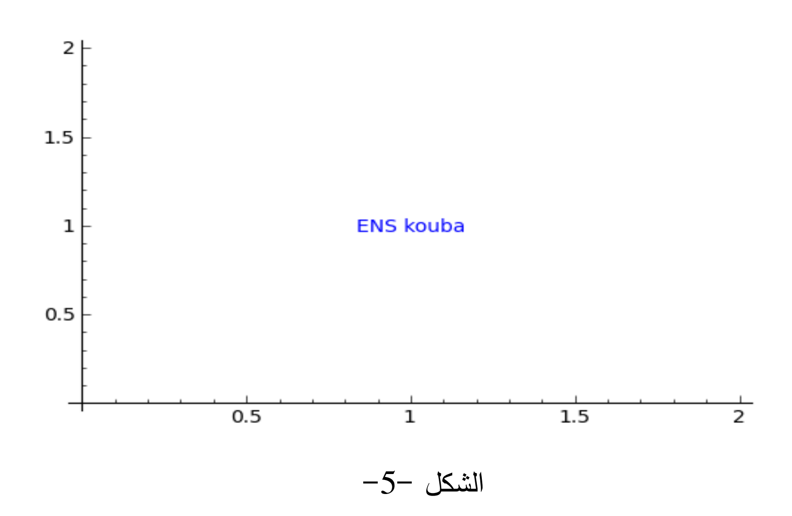
sage: circle((2,1),3,aspect_ratio=1)



الشكل-4- الشكل-3-

3) إدراج نص: من أجل تسمية المحاور أو المنحنى، ولكن لايمكن إعطاء التسمية باللغة العربية text("txt",(x,y),option)

sage: text("ENS kouba",(1,1))



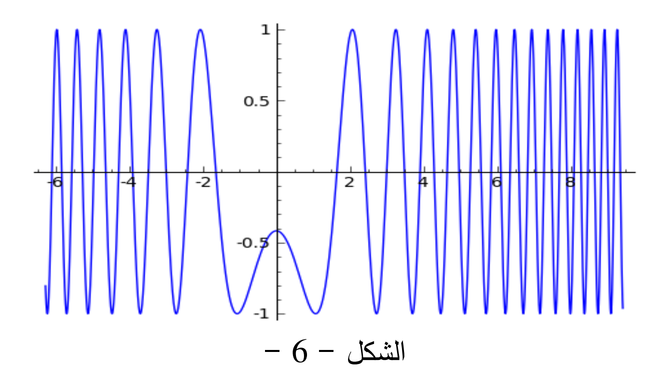
الشكل -5-

4) رسم منحنى دالة:

بواسطة الأمر "plot()" يمكن رسم المنحنيات المميزة للعديد من الدوال، ونستعمل الأمر كالتالي plot(f(x),(x, xmin, xmax),options)

نرسم منحنى الدالة "cos(x^2+2)" من النقطة (−2pi) إلى النقطة (2pi)

```
sage: plot(cos(x^2+2),x,-2*pi,3*pi)
```



الشكل – 6 –

5) خيارات الرسم ثنائي البعد:

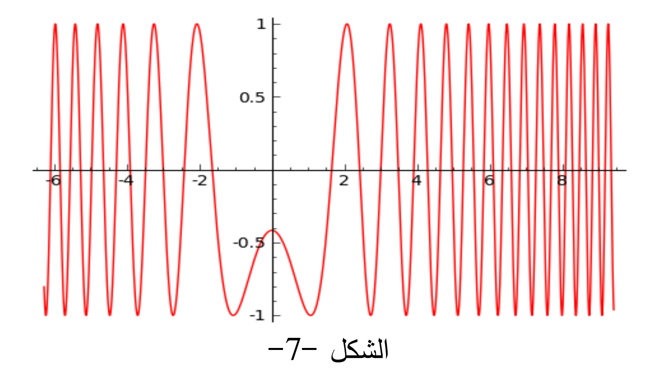
2) خيارات اللون:

ونجد ضمن خيارات الرسم اللون والذي يمكن التحكم به بواسطة الأمر "color" ونكتب اللون الذي نريده للمنحنى مثلا أحمر (red) أزرق (bleu) أخضر (green) أو عن طريق شيفرة الألوان (color='#ff00ff') أو عن طريق قيم الألوان التالية (color=(r,g,b)) حيث

r: أحمر، g: أخضر، b: أزرق، بحيث $0\leq r,g,b\leq 1$

نرسم منحنى الدالة "cos(x^2+2)" من النقطة (-2pi) إلى النقطة (2pi).

```
sage: plot(cos(x^2+2),x,-2*pi,3*pi , color=('red'))
```

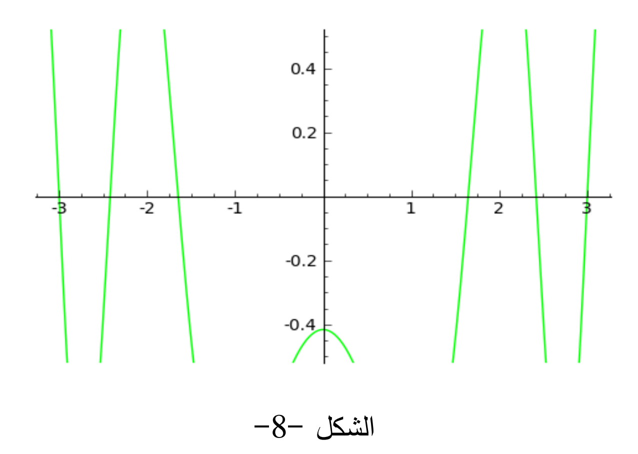


الشكل –7–

3) ويمكن التحكم في طول المحاور، بواسطة القيم العليا والدنيا للمحاور (xmin, xmax, ymin, ymax).

وهذه طريقة ثانية للتحكم بقيم الألوان

```
sage: plot(cos(x^2+2),xmin=-pi,xmax=pi,ymin= .5,ymax=.5,color=(0,1,0))
```

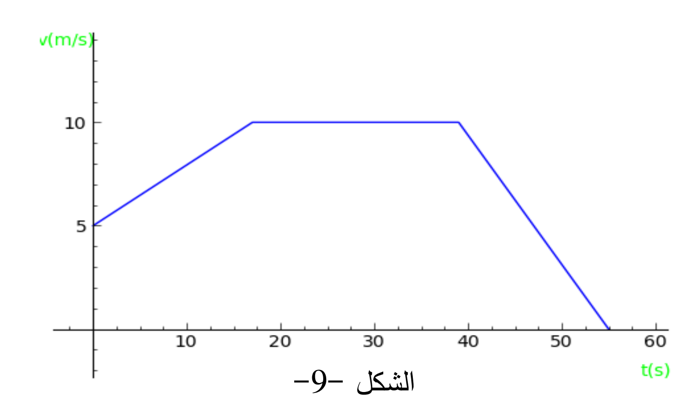


الشكل -8-

4) تسمية المحاور:

رسم منحنى تعليمي: تغيرات سرعة متحرك بالنسبة للزمن

```
sage: f1=line([(0,5),(17,10),(39,10),(55,0)])
sage: f2=text("t(s)",(60,-2),color=(0,1,0))
sage: f3=text("v(m/s)",(-3,14),color=(0,1,0))
sage: plot(f1+f2+f3)
```

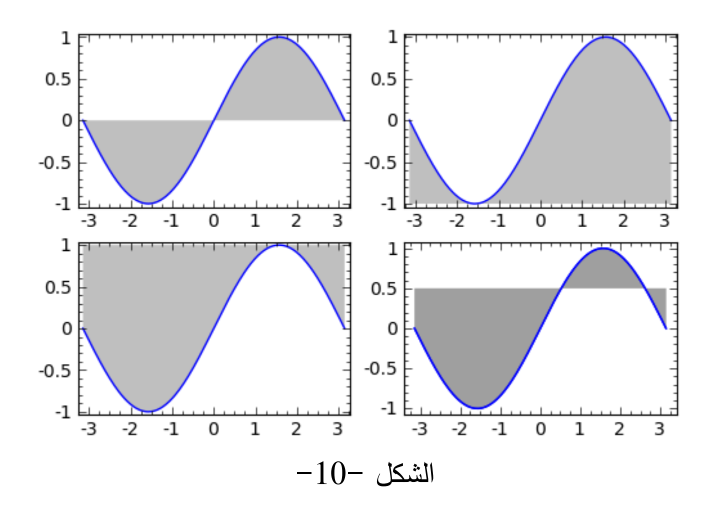


الشكل -9-

5) التظليل:

كما يمكن إضافة تظليل حول الدالة مع عدة إمكانيات للتضليل، وأيضا رسم عدة منحنيات منفصلة ولكن في أمر واحد، وإخفاء المحاور لاحظ المثال التالي:

```
sage: p1 = plot(sin(x), -pi, pi, fill = 'axis')
sage: p2 = plot(sin(x), -pi, pi, fill = 'min')
sage: p3 = plot(sin(x), -pi, pi, fill = 'max')
sage: p4 = plot(sin(x), -pi, pi, fill = 0.5)
sage: graphics_array( [[p1, p2], [p3, p4]] ).show(frame=True, axes=False)
```



الشكل -10-

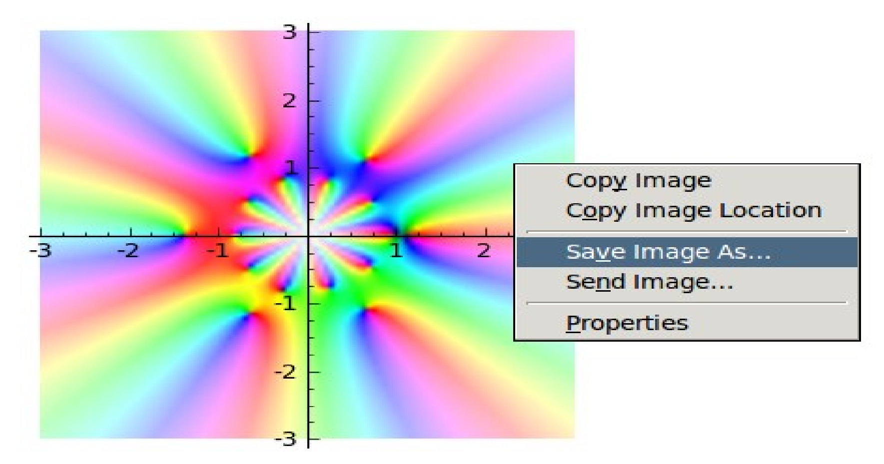
وهذا إضافة إلى خيارات أخرى مثل سمك المنحنى "=thickness" مقياس الشكل ."figsize=[3,3]"

ولكن كيف يمكن حفظ الصور؟

6) حفظ الصور :

لدينا طريقتان لحفظ الصور :

الأولى: بعد ظهور الصورة في متصفح الإنترنت، نضغط بزر الفأرة الأيمن على الصورة ونختار الأمر save image لاحظ الشكل التالي:



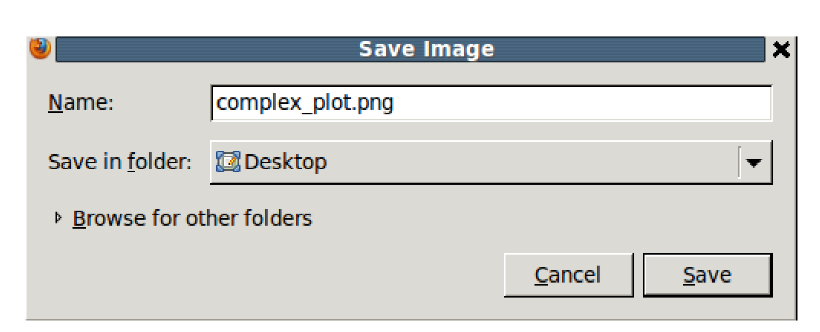
الشكل -11-

ليظهر مربع الحوار التالي:

نسمي الصورة ونختار مكان الحفظ.

الثانية: عن طريق الأمر "save()"

```
sage: f = z^7 + z^2 -5*z+ 1 + 1/z^9+1/z^2
sage: pl=complex_plot(f,(-3, 3), (-3, 3))
sage: pl.save( "complex_plot.pdf" )
```



الشكل -12-

ويوفر برنامج sage الامتدادات التالية كخيارات للحفظ " png, .ps, .eps, .svg, .sobj.."، ويمكن أيضا إضافة خيارات أخرى كالتي ذكرنها سابقا من أجل الحفظ.

ملاحظة: نجد الصورة محفوظة داخل ملف "/home/sage"

كما أن هناك أوامر أخرى للتمثيل البياني ثنائي البعد موجودة في وثيقة Sage Reference v4.6.1، الذي يعتبر المرجع الرئيس للبرنامج باحتوائه في هذا الإصدار على 6202 صفحة.

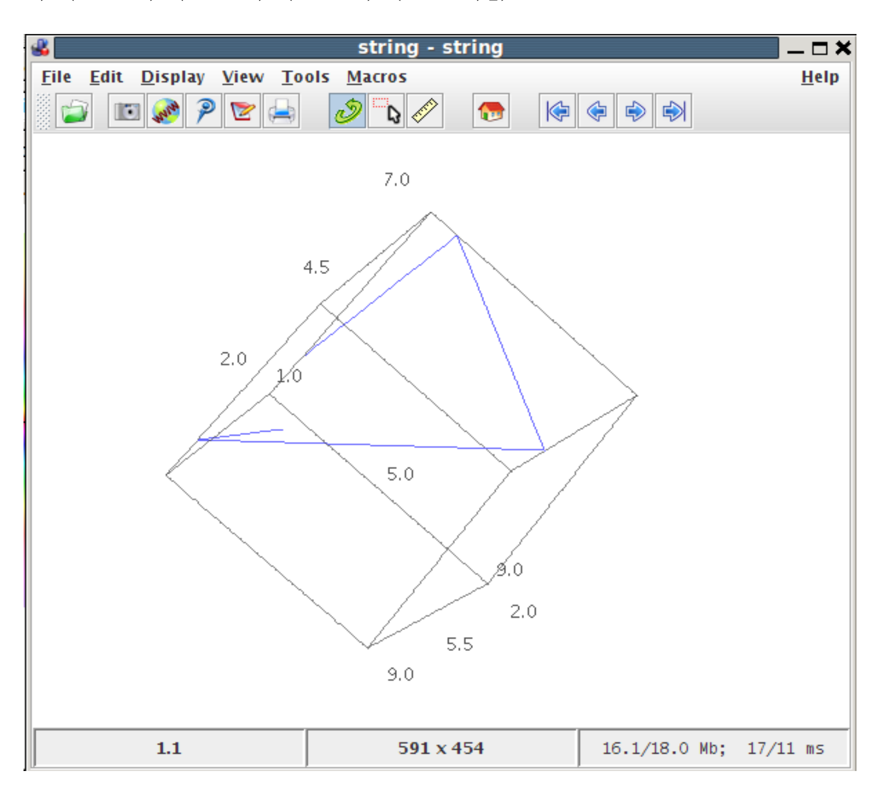
2.X. الرسم ثلاثي الأبعاد:

الرسم ثلاثي البعد في sage في منتهى السهولة ويدعم الكثير من الاختيارات، وهذا الجزء الوحيد من sage الذي يعمل بواسطة برنامج java تحت اسم "jmol".

1) رسم خط ثلاثي البعد:

للرسم في البعد الثلاثي نستعمل الأمر "line3d()" بالطريقة التالية:

```
line3d([(x1,y1,z1),. . .,(xn,yn,zn)],options))(
line3d([(2,3,2),(1,9,3),(9,7,7),(2,2,7),(1,2,3)])
```



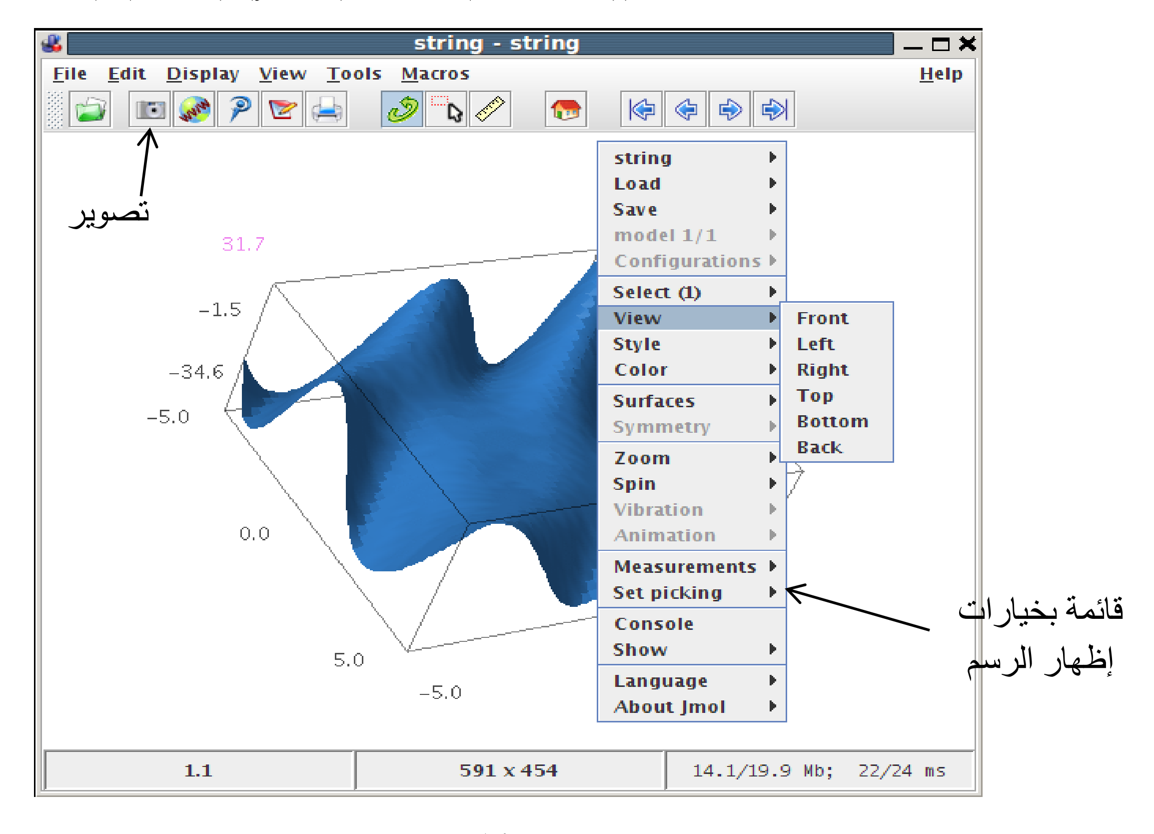
الشكل -13-

يظهر لنا الرسم في نافذة جديدة، تمكن المستخدم من التفاعل معها بكل سهولة، فيمكنه رؤية الشكل من أي زاوية يريدها فقط باستعمال الفأرة والقيام بالتحريك، أو بالنقر بزر الفأرة الأيمن لتظهر مجموعة كبيرة من الاختيارات.

2) رسم دالة ثلاثية الأبعاد:

نرسم الدالة $f(x,y) = x^2 \sin(x+y) + y^2 \cos(x+y)$

```
sage: var('x y')
(x, y)
sage: f=x^2*sin(x+y)+y^2*cos(x+y)
sage: plot3d(f,(x,-5,5),(y,-5,5),color=(0.2,.5,0.8))
```



الشكل -14-

من بين قوائم الأوامر المهمة التي يتيحها لنا برنامج jmol:

قائمة save: حفظ الصور بعدة امتدادات.

قائمة View: بواسطة هذا الأمر نغير زاوية الرؤية.

قائمة Style: كيفية عرض الرسم ومن بين الأوامر التي يحويها إظهار المحاور.

قائمة color: تلوين المحاور، الخلفية...

قائمة spin: التدوير الآلي للرسم، في عدة وضعيات.

كما يمكن الحصول على صورة من الرسم من بالنقر على الأيقونة أعلى النافذة.

3) خيارات الرسم ثلاثي البعد

أما خيارات الرسم ثلاثي البعد لا تختلف كثيرا عن الرسم في البعد الثنائي:

- اللون" color=(r,g,b)"
- السمك "thickness="
- النسب بين المحاور "aspect_ratio=[1,1,1]"
- الشفافية "opacity=n"، صورة ثنائية البعد عن المجسم ثلاثي البعد (viewer="tachyon")...
- تلوين المستويات بألوان مختلفة adaptive = true، إخفاء المحاور" frame = False"...

XI. أداة التحكم التفاعلية interact:

تم إدخال هذا الأمر من طرف "stien willien" كرد على أمر "munipulate" الموجود في برنامج 'mathimatica' وهذا بتاريخ 2008/03/02 ، خلال ملتقى sage الثامن، وقد شارك في تطوير هذا الأمر "Jason Grout".

وبواسطة هذا الأمر يسهل لنا الكثير من العمليات والتجارب، وخصوصا من أجل محاكاة التجارب التي نستعملها في التدريس، عند استعمال هذا الأمر يمكن اختصار الكثير من أسطر الأوامر عبر اختيار أزار أو شريط تمرير للتحكم في المدخلات وبالتالي التحكم في النتائج.

ملاحظة: نستخدم واجهة " notebook"

لاحظ الأمثلة التالية:

نقوم بحساب مجموع عددين من أجل قيم مختلفة، ويتم تغيير قيم a,y بواسطة شريط تفاعلي تتغير قيمة كل من a و y من القيمة 1 إلى 10 بخطوة واحدة.

```
sage: @interact
....: def _(a=(1,10,1),y=(1..10)): print a+y
....:
```

```
a   2
y   2

4
```

الشكل -15-

رسم منحنى حركة اهتزازية معادلتها من الشكل $\sin(ax + b + 1)$ مع إمكانية تغيير قيم a و b بواسطة إدخال القيم في الخلية الخاصة بالمتغير.

```
sage: @interact
... def _(a = input_box(1),b= input_box(0)):
...     show(plot(sin(1+a*x+b), (x,0,6),color=("red")), figsize=5)
```

a 1
b 3

الشكل -16-

تحليل عدد إلى جداء عوامل أولية، ما عليك سوى إدخال العدد ليقوم البرنامج بتحليله، بهذه الطريقة تختصر الوقت لتحليل مجموعة كبيرة من الأعداد.

```
sage: @interact
... def _(t1=text_control("التحليل إلى جداء عوامل أولية"), n="1"):
...     print factor(Integer(n))
```

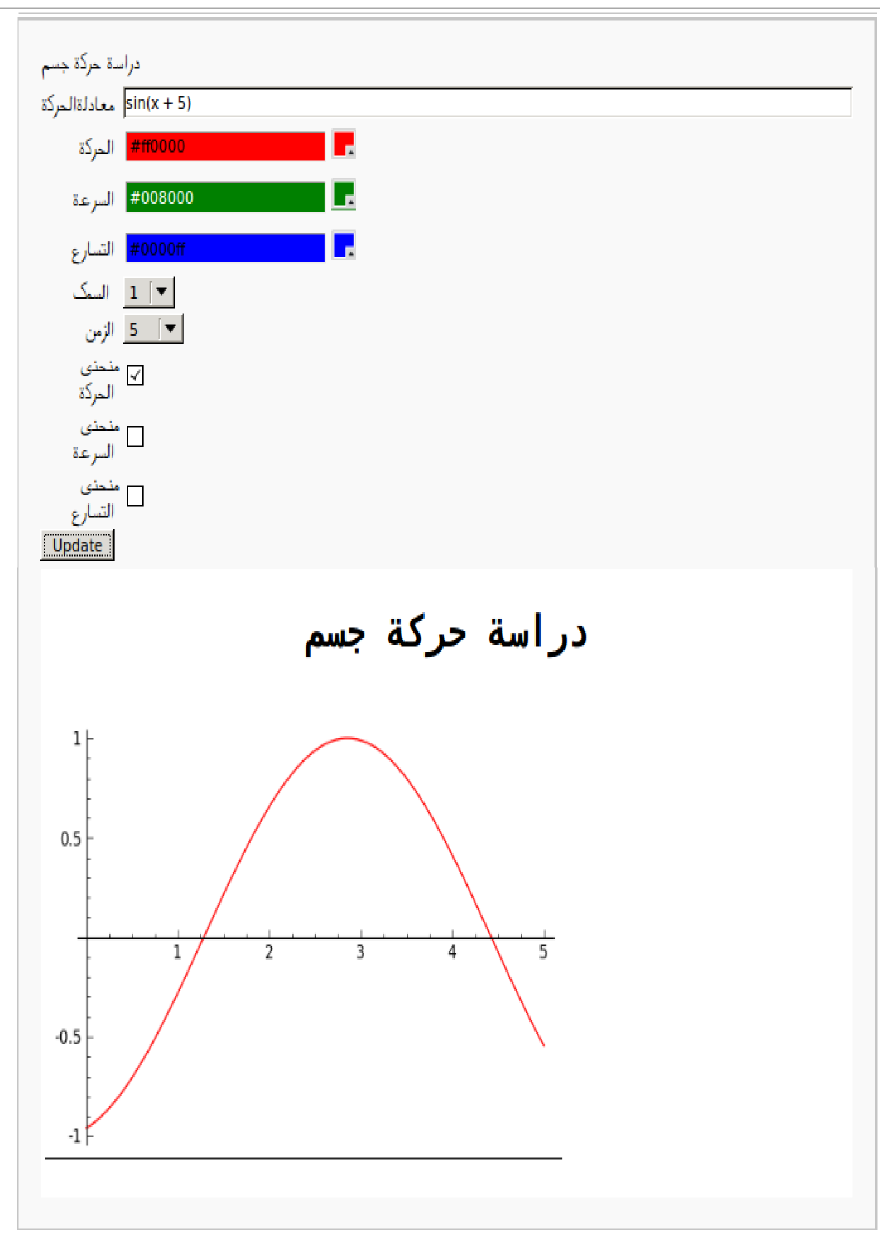
التحليل إلى جداء عوامل أولية
n 25

5^2

الشكل -17-

كما ارتأينا أن نضع هذه المحاكاة البسيطة والتي بواسطتها نستطيع أن نرسم منحنى حركة جسم ومنحنى سرعته، وتسارعه، بعد إدخال معادلة الحركة، مع وجود إمكانية تغيير ألوان وسمك المنحنيات، وأيضا مدة الحركة.

```
@interact
def _(bb=text_control("دراسة حركة جسم"), X=input_box(sin(x+5), 'معادلة الحركة'),
     a=('الحركة',Color('red')),v=('السرعة',Color('green')),vv=('التسارع',Color('blue')), thickness=('السمك',[1..10]),t=('الزمن',[5..100]),axes1=('منحنى الحركة', false),axes2=('منحنى السرعة', false),axes3=('منحنى التسارع', false),auto_update=False):
     html('<h1 align=center>%s</h1>'%bb)
     show(plot(X, 0,t, color=a, thickness=thickness,alpha=axes1)+  plot(diff(X), 0,t,color=v,thickness=thickness,alpha=axes2)+plot(diff(X,2), 0,t,color=vv,thickness=thickness,alpha=axes3))
```



الشكل -18-

والآن نستعرض كيفية إنشاء هذه المحاكاة، وماهي وضيفة الأوامر الموجودة داخلها:

@interact: الأمر التفاعلي أي أن ما سيأتي عبارة عن أوامر للواجهة تفاعلية.

Def_() : إدخال الأوامر التي تتحكم في المتغيرات بين قوسي هذا الأمر وعند الانتهاء نضع نقطتين.

ومن بين هذه الأوامر التي من الممكن إدخالها ضمن هذا الأمر نجد هذه الأوامر التي استعملناها في هذه المحاكاة.

bb=text_control('اسم'): صندوق النصوص، أعطيناه الرمز bb لاستعماله بهذا الرمز لكتابة عنوان المنحنى (نص html الموجود في السطر السادس)، والتسمية التي تظهر في الواجهة التفاعلية هي الموجودة بين الفاصلتين التنصيصيتان.

دراسة حركة جسم

X=input_box(sin(x+5),'معادلة الحركة') : صندوق الإدخال، بواسطة هذا الأمر يمكن إنشاء صندوق حوار ضمن شاشة التفاعلية للمتغيرات، وفي حالتنا هذه استعملناه من أجل إدخال المعادلات، ووضعنا له الرمز 'X' ليقوم البرنامج برسم منحنى المعادلة التي نقوم بإدخالها في صندوق الإدخال، والمعادلة 'sin(x+5)' هي القيمة الافتراضية لهذا الصندوق.

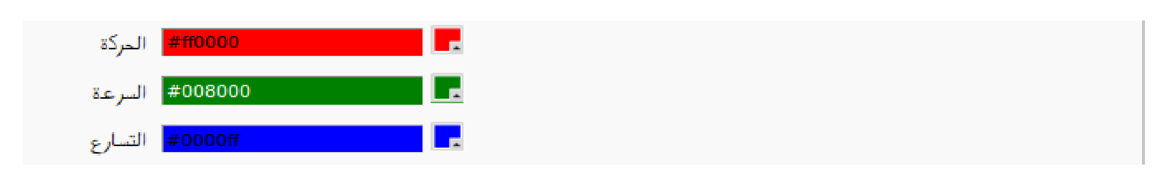
وقد قمنا بتسمينه بـ 'معادلة الحركة' وهي التي تظهر ضمن الشاشة التفاعلية، كما

معادلةالحركة sin(x + 5)

يمكنك اختصارا كتابة هذا الأمر كالتالي ((sin(x+5),'معادلة الحركة')=X.

a= ('الحركة' ,Color('red')) : صندوق تغيير الألوان تحت رمز a لاستعماله في تغيير ألوان منحنى الحركة، واللون الافتراضي له هو الأحمر، والاسم الذي سيظهر على الواجهة 'الحركة'.

vv= ('التسارع',Color('blue'))، v= ('السرعة' ,Color('green')): نفس الكلام السابق عن a.



ملاحظة: يجب كتابة Color بحرف C كبير.

thickness=([10..1],' السـمك'): قائمـة تغييـر السُـمك، رمزهـا thickness، واسـمها 'السمك'.

t=([100..5],'الزمن') : قائمة تغيير الزمن نفس الكلام السابق عن السمك.

السمك 1
الزمن 5

axes1= (false ,' منحنى الحركة') : علبة تأكيد تأخذ القيمة '1' في حالة وضع العلامة 'صـحيح' داخـل المربـع والقيمـة '0' فـي حالـة العكـس، رمزهـا 'axes1' اسـتعملناه مـن أجـل إظهار وإخفاء منحنى الحركة (التحكم في قيم شفافية منحنى الحركة).

axes2= (false ,'منحنـى السـرعة') , axes3= (false ,'منحنـى التسـارع'): نفس الكـلام السابق عن العلبة axes1.

auto_update=false: إيقاف التحديث الآلي، أي أن التحديث لا يتم إلا بعد النقر على أيقونة 'update' الموجودة في الواجهة التفاعلية.

Update

وبهذه الطريقة نكون قد أتممنا كتابة أوامر إدخال المعطيات، والآن ننتقل إلـى توضـيح عمل هذه المحاكاة، وكيفية ربطها مع أوامر المدخلات التي ذكرناها سابقا.

السـطر السـادس 'html('<h1 align=center>%s</h1>'%bb)' مـن الأوامـر يعنـي أن البرنامج يقوم بكتابة النص الموجود في صندوق النصوص 'bb' في رأس منتصف الرسم.

دراسة حركة جسم

show(): هذا الأمر من أجل إظهار العمليات التي تتم ما بين قوسيه.

هذا أمر رسم المعادلة : plot(X, 0,t, color=a, thickness=thickness,alpha=axes1) 'X' التي نقوم بإدخالها في الصندوق النصوص، من أجل قيم المتغير 'x' في المجال (0,t) ويحدد 't' من خلال قائمة 'الزمن' التي رمزها 'a'، ويمكن التحكم بسمك المنحنى من خلال الأمر 'thickness' الموجود تحت اسم 'السمك' في الواجهة التفاعلية، كما يمكن تحدد قيم شفافية المنحنى من خلال الأمر 'alpha' والذي يُظهر المنحنى (لون غامق) عندما تكون قيمته '1'، ويُخفي المنحنى عند تكون قيمته '0'، ونتحكم بهذه القيم بواسطة مربع التأكيد الموجود برمز 'axes1' وباسم 'منحنى الحركة' في الواجهة التفاعلية.

plot(diff(X), 0,t,color=v,thickness=thickness,alpha=axes2)+plot(diff(X,2), 0,t,color=vv,thickness=thickness,alpha=axes3))

السطران أعلاه يمثلان أمر رسم المشتق الأول للدالة 'X'، والمشتق الثاني لها، ونفس ماقيل في رسم المعادلة 'X' ينطبق على هذين السطرين.

وبهذا نكون قد انتهينا من شرح كيفية كتابة المحاكاة، وفيما يلي نعطي قائمة بأوامر أخرى للتحكم في المدخلات.

n = slider(4,50,1) : شريط تمرير في المجال (4,50) بخطوة قيمتها '1' .

u 22

a = range_slider(-100,50,2): مجال التغير بقيمتين لتحديد المجال، مجاله من −100 إلى 50 بخطوتين.

a (31, 67)

s = checkbox(): مربع التأكيد قيمته الافتراضية تساوي '1'.

s ☑

s = selector((1,10,5,59,74,2,41,96,87,48), buttons=true): قائمة بمجموعة من الاختيارات التي من الممكن وضعها في أزرار.

s 1 10 5 59 74 2 41 96 87 48

اختيار :e = color_selector (0,0,1), widget='colorpicker', hide_box=true) الألوان مع إمكانية تغيير علبة الألوان 'widget' إلى 'jpicker' أو 'farbtastic' ، وإخفاء مستطيل اللون الافتراضي.



إدخال مجموعة :r = input_grid(2,8,[0,2,0,7,1,9,8,8,3,0,0,3,2,0,1,1],width=4) من المتغيرات على شكل مصفوفة (مجموعة من الخلايا) ، بسطرين و 8 أعمدة وحجم كل

r	0	2	0	7	1	9	8	8
	3	0	0	3	2	0	1	1

خلية يساوي 4 (مثلا).

كل أمثلة الأمر "interact" الموجودة في هذا الجزء، وأخرى تجدونها ضمن القرص المرفق مع المذكرة تحت اسم "نماذج تفاعلية" لذا يرجى تحميلها إلى البرنامج في جهاز الكمبيوتر، أو الاطلاع عليها في واجهة البرنامج على الانترنت عبر الموقع sagenb.org

XII. البرمجة:

برنامج sage كما قلنا برنامج ثري بأوامره، ولكنكم أيضا تستطيعون كتابة الأوامر ، أو اللغرتميات الخاصة بكم، سنعطيكم لمحة فقط عن هذه العملية لنترك الباقي لكم عند مطالعة وثائق sagemath الممنوحة من طرف مطوري هذا البرنامج لكي يطوروا قدراتكم في استعمال هذا البرنامج، وأيضا أفكار قد تساعدكم في الرياضيات.

بواسطة هذا المثال نقوم بتسمية أمر " puiss2 (u, k) " الذي يقوم بحساب u^k.

```
sage: defpuiss(u, k) :
....:     v=1
....:     whilek!=0:
....:        if k%2==0: u=u*u;k=k/2
....:        else : v = v*u ; k = k-1
....:     return v
....:
sage: puiss(2, 10)
1024
```

الخلاصة:

من خلال هذا الفصل تعرفنا على بعض الأوامر في برنامج sagemath، رغم عدم خبرتنا في برامج من هذا النوع إلا أن بساطة البرنامج، والوثائق المتوفرة لديه من مواقع إلكترونية و وثائق pdf، ودروس مصورة جعلت من العمل به بداية شيقة.

وعند التعامل مع البرنامج، نجد أن الأوامر التي يستعملها لا تختلف كثيرا على الأوامر المعترف بها في باقي برامج الرياضيات الأخرى، وهذا حتى يسهل الانتقال من هذه البرامج التجارية إلى البرامج المجانية.